

Bitwise operations

Bitwise operators perform operations directly on the binary representation of numbers.

ex of bitwise AND(&), OR(|), and NOT(~) operators.

how these bitwise operators work:

- Bitwise AND(&): compares each bit of two numbers. the result is 1 only if both corresponding bits are 1.
- Bitwise OR(|): compares each bit of two numbers. the result is 1 if at least one of the corresponding bits is 1.
- Bitwise NOT(~): A unary operator that inverts all the bits of a single number changing 1s to 0s and 0s to 1s.

1) $4 \& 2$.

The number 4 in binary is 100

The number 2 in binary is 010

$$100 \& 010 \Rightarrow 000$$

bitwise $4 \& 2 = 0$

2) ~ 4

$$\sim 4 = -(4+1)$$

$$= -5$$

3) $4 | 2$

4 in binary 100

2 in binary 010

$$4 \rightarrow 100$$

$$2 \rightarrow 010$$

$$100 | 010 \Rightarrow 110_2$$

$$(1 \times 2^2) + (1 \times 2^1) + (0 \times 2^0) = 4 + 2 + 0 = 6$$

~~the bit~~

4) $4 / 2$

The operation $4 / 2$ is a standard division

the result of division is 2.

⑤ Relation operators

Relational operators compare two values and return a boolean result: 1 for true and 0 for false.

operator	Example	result
> (greater than)	$4 > 2$	1 (True)
< (less than)	$4 < 2$	0 (False)
>=	$4 >= 2$	1
<=	$4 <= 2$	0
=	$4 == 2$	0
!= (not equal)	$4 != 2$	1

C program for relational operators

```
#include <stdio.h>

int main() {
    int a = 4;
    int b = 2;

    printf("4 > 2 is : %d\n", a > b);
    printf("4 < 2 is : %d\n", a < b);
    printf("4 >= 2 is : %d\n", a >= b);
    printf("4 <= 2 is : %d\n", a <= b);
    printf("4 == 2 is : %d\n", a == b);
    printf("4 != 2 is : %d\n", a != b);

    return 0;
}
```